محمد امانلو 810100084

B : NP-hard , $A \le_p B$ A : NP-complete

۱) الف) نادرست

مسئله ای NP-complete است که از دو کلاس NP-hard و NP باشد اما درباره مسئله A نمی توانیم بگوییم دقیقا از کدام یکی از این دو است. زیرا وقتی مسئله A به B کاهش پیدا کند نمی توان نتیجه گرفت که هر مسئله NP به A کاهش پیدا کند یا چک کردن جواب A با اردر چندجمله ای انجام می شود.

برای مثال می توانیم A را یک زبان که از کلاس NP است در نظر بگیریم که NP-hard نیست زیرا B، NP-hard است و داریم $A \le_p B$ اما A، NP-complete نیست.

A : NP-complete $A \le_p B$ B : NP-complete

ب) نادرست.

فرض می کنیم B یک مسئله NP-hard باشد که NP نیست حال $A \le_p B$ درست است اما B، NP-complete نیست.

ج) درست تمام مسائل NP-complete، NP هم هستند. و مسائل NP را می توان در زمان چندجمله ای به NP-comp کاهش داد پس یک مسئله NP-complete را می توانیم در زمان چندجمله ای به یک مسئله NP-complete دیگر کاهش دهیم.

د) درست. فرض می کنیم B یک زبان دلخواه از کلاس NP باشد از آنجایی که P = NP است پس B یک deterministic decider با زمان حل چندجمله ای دارد. فرض می کنیم نام آن D باشد همچنین فرض می کنیم A زمانی که ورودی w_a باشد accept و زمانی که ورودی w_r باشد reject می کند.

به ازای ورودی w به B، ابتدا D(w) را اجرا می کنیم اگر accept کرد w_a را به A می دهیم در صورت reject، w_r را به A می دهیم. همچنین داریم $w \in L_B \iff f(w) \in L_A$. D(w) در زمان P تصمیم می گیرد پس تابع f نیز در زمان P است پس $B \le_p A$

```
w ---> f
       D   decider of B
       D'  decider of A
       run D(w):
         if (accept):
             run D'(w_a)
         else:
             run D'(w_r)
                        ---> accept
                        ---> reject
```

②

$$Mode P = \{ \langle a,b,c,p \rangle \mid a^b = c \pmod p \}$$

فرض می‌کنیم تعداد بیت‌های عدد b برابر n_b باشد. عدد b را به صورت جمع چند عدد باینری با تنها یک بیت 1 می‌نویسیم. ($b = b_1 + b_2 + b_3 + \dots + b_n$) حال داریم. $a^b = a^{b_1+b_2+b_3+\dots+b_n} = a^{b_1} a^{b_2} \dots a^{b_n}$

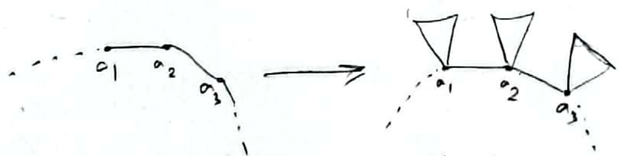
n_b عدد ایجاد می‌شود که حداکثر برابر b است. هر کدام از a^{b_i} ها را طبق راهنمایی ($a^{(1000)_2} = ((a^2)^2)^2$) می‌نویسیم.

این کار بر روی ماشین تورینگ $n_a^2 \log_{b_i}$ زمان می‌برد که یک چند جمله‌ای است.

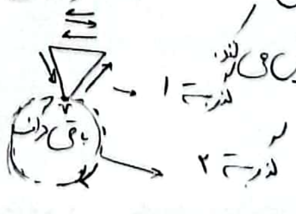
n عدد داریم که باید حاصل ضرب آنها را حساب کنیم. این کار در O(polynomial) انجام می‌شود در نتیجه Mode P در O(polynomial) حل می‌شود و در دسته P قرار دارد.

③ ← مسئله تشخیص وجود دور "گذر دوبل از رئوس" در گراف G، Np-hard است.

یک راس دلخواه v در گراف اصلی را انتخاب می‌کنیم و به ازای آن دو راس v' و v" را اضافه می‌کنیم. و بین این ۳ راس یال‌هایشان را رسم می‌کنیم. با این کار هر راس در گراف اصلی در گراف جدید تبدیل به یک راس برشی می‌شود.

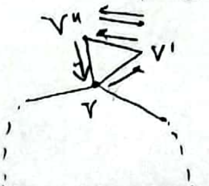


حال اگر گراف جدید گذر دوبل داشته باشد. از هر راس v در گراف اصلی دوبار گذر کرده‌ایم. و از آنجایی که v راس برشی است کل گذر را به ۲ گذر بسته جدا تبدیل می‌کند.



اگر به ازای هر راس در گراف اصلی قسمت دورشان با راس‌های v' و v" را حذف کنیم. به یک دور در گراف اصلی می‌رسیم و از آنجایی که به ازای هر راس یکی از گذرهای آن را حذف کردیم از هر راس یک بار گذر کرده‌ایم و دور همیلتونی داریم.

اگر گراف اصلی دور همیلتونی داشته باشد در گراف حاصل گذر دوبل خواهیم داشت زیرا به هر راسی که می‌رسیم یک دور به شکل زیر می‌زنیم و سپس به راس بعدی در دور همیلتونی می‌رویم.



$v \to v' \to v'' \to v' \to v'' \to v$

ادامه (٤) پس گراف اصلی دور همیلتونی دارد اگر و فقط اگر گراف حاصل گذر دوبل داشته باشد. و تبدیل گراف اصلی به گراف حاصل
از اردر تعداد رئوس گراف می باشد زیرا به ازای هر راس، 2 راس و 3 یال اضافه می کنیم پس دور همیلتونی به گذر دوبل کاهش
می یابد پس گذر دوبل Np_hard است. Np-complete بودن دور همیلتونی را فرض کردیم

④ مسئله setcover را به مسئله پهلوانان تبدیل می‌کنیم. داریم:

ورودی لیست پهلوانان = مجموعه universal در setcover

لیست اعضای باشگاه‌ها = مجموعه S که شامل زیر مجموعه‌های universal است

لیست باشگاه‌ها = مجموعه‌ای از اعداد ۱ تا $n(S)$

$$K_{\text{پهلوانان}} = n - K_{setcover}$$

در K پهلوانان دنبال آن هستیم که بفهمیم با نگه داشتن K باشگاه می‌توان به جواب رسید یا نه و در setcover می‌خواهیم بفهمیم با حذف K مجموعه می‌توان به جواب رسید یا نه.

اگر مسئله پهلوانان حل شود setcover نیز حل می‌شود، و اگر setcover حل شود پهلوانان حل می‌شود. پس setcover $\leq_p$ پهلوانان

برای چک کردن جواب پهلوانان تنها می‌بایست چک کرد هر پهلوان حداقل در یک باشگاه است پس P است.

پس مسئله پهلوانان NP و NP-hard است پس NP-complete است.

⑤ الف) فرض می‌کنیم در گراف اصلی راس دلخواه v به t (راس مقصد) متصل می‌باشد. تغییرات زیر را در گراف اعمال می‌کنیم:

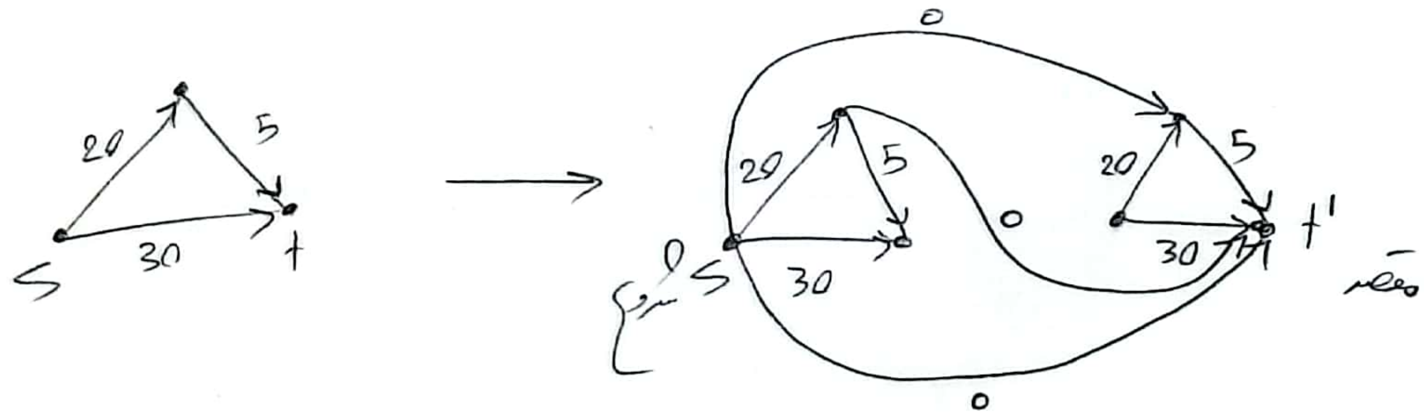
راس t' را به گراف اضافه می‌کنیم و همان یال v به t را حذف می‌کنیم و با همان وزن v به t' را رسم می‌کنیم. از t' به t یالی با وزنی بزرگتر از ماکسیمم وزن گراف اصلی رسم می‌کنیم. به این شکل حل کننده B مجبور است آن یال را صفر کند زیرا به ازای هر مسیر، پرهزینه‌ترین یال همان یال $t'-t$ است پس جواب B بدون در نظر گرفتن $t'-t$ همان مسیر min در گراف اصلی است که حل کننده A خروجی می‌دهد. تبدیل این گراف به گراف ورودی B از اردر خطی است زیرا max به اندازه رئوس، یال اضافه می‌کنیم.

5) ب) یک گراف کپی شده گراف اصلی می سازیم اگر راس v در گراف اصلی باشد متناظر آن در گراف کپی شده v' است.

اگر در گراف اصلی از راس a به b یال داشته باشیم از a به b' یالی با وزن صفر رسم می کنیم. با این کار بین گراف اصلی و گراف کپی شده یال های با وزن صفر وجود دارد. که جهت آنها از اصلی به کپی شده است پس اگر این مجموعه را به حل کننده A بدهیم می تواند تمام مسیرهای را که وزن یکی از یال ها صفر در نظر گرفته شده، هم بررسی کند پس می تواند مسیری که B خروجی می دهد را پیدا کند.

در گراف نهایی مقصد t' و شروع s است.

این کاهش خطی است زیرا گراف را فقط کپی کرده ایم. و بین دو گراف به اندازه تعداد یال های گراف اصلی یال می کشیم. پس اردر تبدیل در اردر تعداد یال ها است.



⑥ مسئله 3-SAT را به این مسئله کاهش می‌دهیم:

فرض می‌کنیم 3cnf به ما داده شده است S را می‌سازیم طوری که: $S=\{x_1,\bar{x}_1,\dots,x_m,\bar{x}_m\}$

به ازای هر clause ای که در 3cnf داریم یک مجموعه 4 تایی تشکیل می‌دهیم که 3 عضو آن clause به همراه y داشته باشد (y جزو 3cnf داده شده نیست)

این مجموعه‌های ساخته شده را در X قرار می‌دهیم و به X مجموعه‌های $\{x_1,\bar{x}_1\}$ ، $\{x_2,\bar{x}_2\}$ ، .... ، $\{x_m,\bar{x}_m\}$ را اضافه می‌کنیم

دو مجموعه S و X را به عنوان ورودی به مسئله می‌دهیم.

می‌توان گفت اگر جوابی برای آن پیدا شود، آنهایی که رنگ یکسانی با y دارند را false و آنهایی که رنگ یکسانی با y ندارند را true بگیریم، جواب مسئله 3sat هم پیدا می‌شود.

حال می‌توان گفت اگر جوابی برای sat پیدا شود آن literal هایی که درست هستند را قرمز کنیم و آنهایی که false هستند را آبی کنیم و y را هم آبی کنیم جواب این مسئله نیز می‌باشد. چرا که $\{x_i,\bar{x}_i\}$ ها که یک آبی و یک قرمز دارند و از آنجایی که مسئله 3sat حل شده، حداقل یک true در هر قسمت هست یعنی حداقل یک قرمز در هر مجموعه و از آنجا که y در همه مجموعه‌ها هست و آبی می‌باشد یعنی در همه مجموعه‌ها حداقل یک قرمز و آبی را داریم. پس این مسئله np-hard است.

از آنجایی که چک کردن هر جواب در اردر چندجمله‌ای می‌باشد زیرا باید m مجموعه که در مجموع n عنصر دارند را چک کنیم

پس مسئله np-complete است.

⑦ مسئله پارتیشن یا همان بخش بندی می تواند به مسئله زیر کاهش یابد:

به طوری که اگر فرض کنیم مجموعه S می تواند پارتیشن شود آنگاه آن ها از مجموعه مستطیل های به فرم $\{1 \times (4.x) \mid x \in S\}$ بتوانند تبدیل به

مستطیل های به فرم $2 \times (\sum_{x \in S} 2x)$ شوند یا در واقع بتوانند سطح مورد نظر ما را بپوشانند

در این حالت مستطیل ها از بالا به یک مجموعه و از پایین مجموعه دیگر را نمایش می دهند و بیانگر مجموعه ها در مسئله اول هستند

همچنین ضرب در 4 انجام شده جهت اطمینان از این است که همه مستطیل ها به صورت افقی در مساحت قرار دارند حال چون این کاهش

انجام شدنی است و با توجه به اینکه می دانیم مسئله A، Np-complete است می فهمیم مسئله B در دسته مسائل Np hard قرار دارد.

ادامه (7)

فرض می‌کنیم مجموعه T ورودی مسئله A می‌باشد برای این کاهش به ازای هر عدد موجود در T ($x \in T$) کاشی‌ای به عرض 1 و طول $4x$ می‌سازیم

فرض می‌کنیم مجموع اعداد مجموعه T برابر X است.

حال مجموعه کاشی‌های ساخته شده را به همراه اتاق با طول $\frac{4X}{2}$ و عرض 2 به مسئله B می‌دهیم. می‌توان نشان داد که اگر مسئله A ورودی T را بپذیرد مسئله B هم ورودی ساخته شده را می‌پذیرد. زیرا کوچک‌ترین طول کاشی موجود می‌تواند 4 باشد بنابراین آن را به صورت عرضی نمی‌توان قرار داد. اگر بتوان کاشی‌ها را چید یعنی دو سطر به عرض یک کاشی پر شده است که متناظر با جوابی است که برای تقسیم مجموعه به دو گروه با جمع مساوی است.

بنابراین یک $A \leq B$ را توصیف کردیم و از آنجایی که A NP-comp است B هم NP hard است

تبدیل هم تبدیل خطی است زیرا اگر n عدد در T داشته باشیم تنها باید یک مجموعه n تایی کاشی تولید کنیم که $O(n)$ می‌باشد.